STP 1644, 2022 / available online at www.astm.org / doi: 10.1520/STP164420210133

Charles Snyers,[1] Julien Ertveldt,[1] Jorge Sanchez-Medina,[1]
Zoé Jardon,[1,2] and Jan Helsen[1]

Prediction of Melt Pool Temperature for Directed Energy Deposition Using Supervised Learning Methods on Optical Measurement Data

Citation

C. Snyers, J. Ertveldt, J. Sanchez-Medina, Z. Jardon, and J. Helsen, "Prediction of Melt Pool Temperature for Directed Energy Deposition Using Supervised Learning Methods on Optical Measurement Data," in *Progress in Additive Manufacturing 2021*, ed. N. Shamsaei, N. Hrabe, and M. Seifi (West Conshohocken, PA: ASTM International, 2022), 59–73. http://doi.org/10.1520/STP164420210133[3]

ABSTRACT

One of today's ongoing challenges in directed energy deposition (DED) is controlling the geometry and material properties of parts. The objective of this paper is to investigate the relationship between several printing parameters of DED (laser power, laser speed, powder feed rate) and the melt pool temperature. Because DED is a complex and nonlinear process, well-established supervised-learning models such as support vector regression and artificial neural networks are particularly well suited to represent it. The MiCLAD machine, designed at the Vrije Universiteit Brussel, is equipped with a hyperspectral camera that monitors the light emitted at several wavelengths by the melt pool during the building process. A steady-state data set produced by the hyperspectral camera is

Manuscript received December 17, 2021; accepted for publication May 11, 2022.
[1]Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels, Belgium C. S. https://orcid.org/0000-0001-8737-8775, J. E. https://orcid.org/0000-0002-9012-3340, J. S.-M. https://orcid.org/0000-0002-9903-7131, Z. J. https://orcid.org/0000-0002-5028-0480, J. H. https://orcid.org/0000-0002-6574-7629
[2]von Karman Institute for Fluid Dynamics, Waterloosesteenweg 72, 1640 Sint-Genesius-Rode, Belgium
[3]ASTM *International Conference on Additive Manufacturing (ICAM 2021)* held November 1–5, 2021 in Anaheim, CA.

postprocessed by an advanced temperature estimation method, and the limitations of the temperature estimation method are identified and discussed. The temperature data are used as training data for supervised-learning methods, and a study is conducted to compare the performance of the considered methods using the measured optical data. This study demonstrates that the melt pool temperature of the DED process can be effectively modeled through the printing parameters thanks to supervised-learning methods.

Keywords
additive manufacturing, directed energy deposition, machine learning, in situ monitoring, multicolor pyrometry

Introduction

DIRECTED ENERGY DEPOSITION

Directed energy deposition (DED) is a metal additive manufacturing (AM) technique that uses a laser to melt and locally fuse blown metallic powder.[1] It is a promising manufacturing technique for complex parts in demanding domains, such as the aerospace and automotive industries. However, as in other metal AM techniques, there is a complex interplay of several simultaneous physical phenomena.[2,3] Therefore, there are many printing parameters that have a first-order influence on the process, such as laser power, laser speed, powder feed rate, carrier gas flow, and so on. Because of this interplay, DED suffers from a lack of process reproducibility and reliability that currently prevents its wide adoption in mass production. To move past this hurdle, optimization of the process parameters and control of the process are required.[4]

To gather in situ information about the process and optimize it, different sensing strategies have been undertaken. The temperature distribution and geometrical properties of the melt pool were shown to correlate well with the geometrical properties of parts manufactured with DED.[5] Given the high temperature in the melt pool, a contactless method is required. Optical measurement of the emitted light by the melt pool allows the estimation of the temperature by using Planck's law and an estimation of the material emissivity.[6]

DATA-DRIVEN ALGORITHMS

Given the high number of process parameters and the inherent nonlinearities of the DED process, it is not trivial to build a robust model of the process. There has been a lot of effort to create numerical physics-based models, but their accuracy is limited by the approximative knowledge of material properties[7] and by the difference in scale between the local melt pool dynamics and the part.[2]

Data-driven algorithms are good tools for building a model of the process without underlying knowledge of the melt pool dynamics and material properties. Both shallow and deep data-driven models have been applied successfully to DED

in situ monitoring data from various sensing sources.[8–13] Postprocessing of high-dimensional data from optical sources is required most of the time, and such post-processing brings another layer of complexity. Zhang et al.[12] applied the XGBoost and long short-term memory algorithms on transient temperature DED data to predict melt pool temperature but with limited focus on how the temperature was computed from infrared data.

Based on past work, it seems clear that there is a vast variety of data-driven algorithms currently available to researchers, but there is no available method yet to determine a priori which will be the most adapted for a given application or process. However, the amount of data required to successfully train an algorithm differs greatly between shallow and deep models. The data requirement also depends on the inherent complexity of the relationship to be modeled. Snyers et al.[14] conducted a preliminary study on simulated data in which they showed that an order of magnitude of 100 samples seems satisfactory to train shallow supervised-learning (SL) algorithms but that deep algorithms most likely require more data. The present paper collects a sufficiently large experimental data set and trains the same SL algorithms with it.

In conclusion, this paper presents an experimental in situ monitoring data set that was collected in a DED machine and shows how the data set was used in conjunction with SL techniques to produce a model of the steady-state DED process. The first section describes the experimental setup, the process parameters, and a method for extracting temperature from the raw data. In the second section, the postprocessed data set is used to train several data-driven algorithms, and their performance is compared.

Experimental Study

METHODS AND MATERIALS

DED Machine

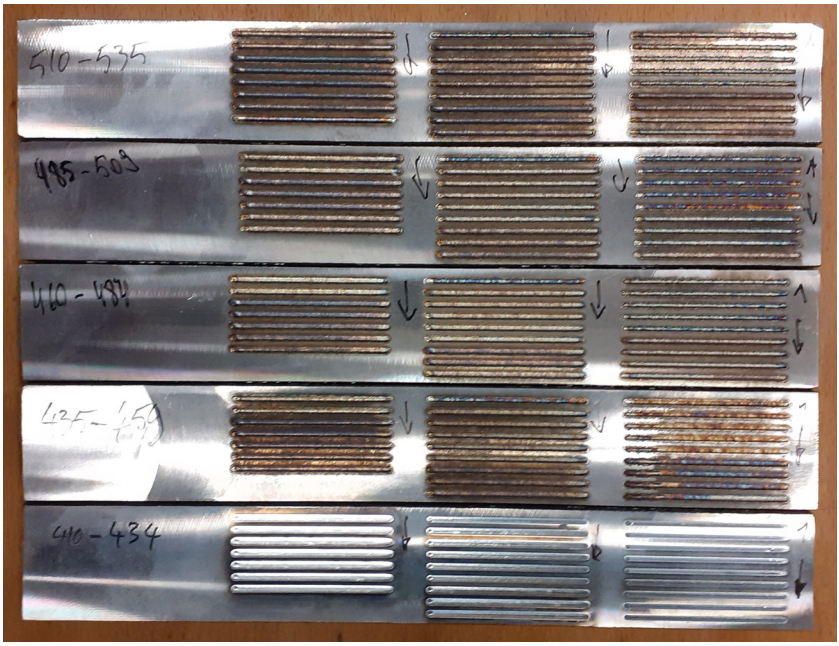
The experimental setup that is used to produce the raw in situ monitoring data is the MiCLAD research platform. It is a DED machine designed and built at the Vrije Universiteit Brussel[15] and is equipped with several sensors, including a multispectral 3D-One Avior AX-M25NIR camera installed coaxially. The camera uses a hyperspectral sensor that is active in the near infrared region (NIR) and provides relevant information to infer the temperature of the melt pool as explained in the next section. A Fabry-Perot filter is deposited on each pixel in a 5×5 mosaic pattern. Therefore, each macropixel captures one value for each of the 25 different bands. The main characteristics of the hyperspectral camera are summarized in table 1.

The experiment consists of printing 126 linear tracks of 40 mm with 316L steel powder (see fig. 1). For each track, the three main process parameters (the laser power P, the laser speed v, and the powder feed rate $\dot{m}$) are varied. To choose the

TABLE 1 3D-One Avior AX-M25NIR camera's main characteristics

Maximum bitlength	8 bits
Full resolution	2,048 × 1,088 pixels
Filter layout	5 × 5
Spectral range	600–1,000 nm
Resolution per spectral band	409 × 217 pixels
Maximum framerate at full resolution	120 fps
Wavelength peaks of Fabry-Perot filter in the NIR	690, 703, 714, 730, 731, 742, 742, 755, 771, 784, 794, 810, 820, 833, 848, 858, 870, 884, 895, 904, 918, 926, 936, 945, and 954 nm

FIG. 1 Tracks printed on the MiCLAD machine.



values of the considered process parameters, the usual energy density (E_d) equation[16] is first modified to take into account the powder feed rate:

$$E_d = \frac{P}{v \cdot d^2 + \dot{m}/\rho} \tag{1}$$

where:

d = laser diameter and

ρ = powder material density.

The values of the process parameters were chosen to keep the energy density between 5 and 30 J/mm^3, which is an acceptable range of parameters for the DED process of 316L steel[17]:

- P: 500, 600, 700, 800, and 900 W
- v: 800, 900, 1,000, 1,100, and 1,200 mm/min
- $\dot{m}$: 0, 1, 2, 3, 4, and 6 g/min

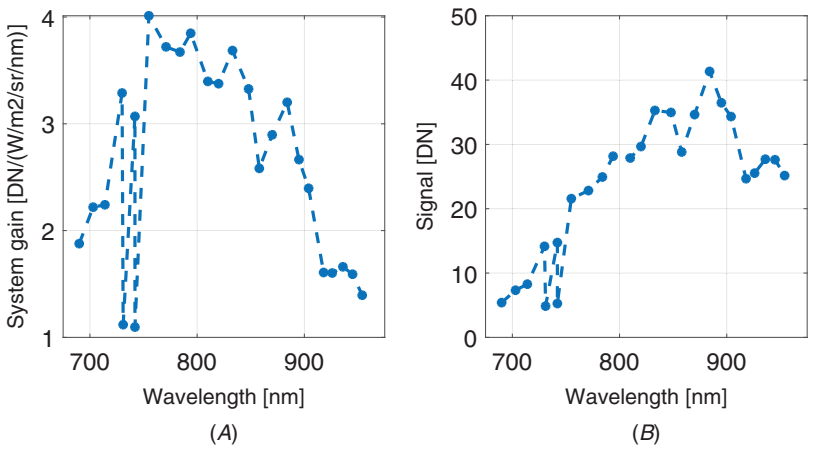
The resulting tracks were printed on stainless-steel strips (see fig. 1). During the printing of each track, the hyperspectral camera captures the light emitted by the melt pool and averages it in the time domain to get the steady-state value of each pixel of the region of interest (the melt pool). In summary, the experiments link each process parameter vector $(P,\ v,\ \dot{m})$ with a hyperspectral image of the melt pool.

Camera Calibration

The gain of the camera was obtained in a calibration experiment in which an integrated sphere (OL Series 455) illuminated the hyperspectral camera through the DED machine optics. Because the spectral radiance of the calibrated lamp is known, the gain can be computed by dividing the mean value of the captured image by the spectral radiance of the lamp at the peak wavelength of the corresponding band. It is assumed that the gain behaves linearly with the camera exposure time Δt. Figure 2 shows the resulting camera gain with respect to band peak wavelength.

To link the pixels to a physical length, a target of known size was placed in front of the optics of the DED machine and was captured by the hyperspectral camera. Thanks to the known size of the target, the size of one pixel could be retrieved (11.36 µm/pixel).

FIG. 2 System gain (*A*) and signal of ideal 1,670-K blackbody system measured with camera (*B*).



POSTTREATMENT DATA

Multicolor Pyrometry

To estimate the temperature $\hat{T}$ from the spectrum $L_{\lambda,i}$ obtained by the camera, the multicolor pyrometry method[18] is used. It comes down to solving the following minimization problem:

$$\hat{T} = \arg\min_{T} \sum_{i=1}^{N} \left| \frac{L_{\lambda,i}}{L_{\lambda,n}} - \frac{B(\lambda_i, T)}{B(\lambda_n, T)} \right| \tag{2}$$

where:

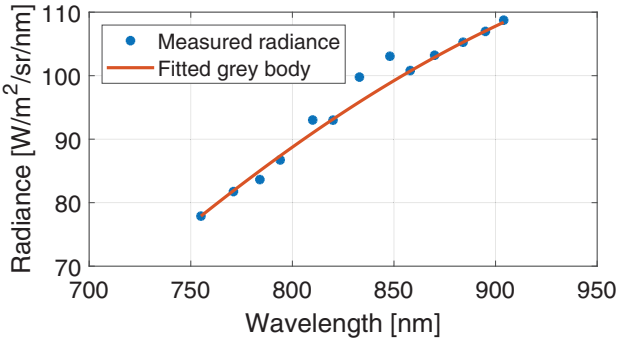
B = spectral radiance of an ideal black body.

The minimization is equivalent to fitting the blackbody equation to the measured spectrum with the least absolute deviations method, using the emissivity as an adjustment variable. The choice of reference wavelength λ_n obviously influences the resulting temperature because it anchors the blackbody equation to a particular $(\lambda_n, L_{\lambda_n})$ pair. The most sensible choice for the reference wavelength is to choose the wavelength at which the spectral radiation measured by the camera has the highest signal-to-noise ratio (SNR).

If the blackbody spectrum at the melting temperature of 316L steel (1,670 K) is multiplied by the system gain, the maximum signal is observed at wavelength $\lambda = 884$ nm (see fig. 2). Therefore, this wavelength was chosen as the reference in the temperature estimation method.

The error to be minimized is influenced by the noise of the measured signals. As can be seen in figure 3, the signals from wavelengths below 754 nm are much weaker than signals from higher wavelengths, and therefore their SNR will be much worse. To limit the influence of noise on the resulting temperature, we decided to ignore signals from wavelengths under 754 nm. Signals from wavelengths above

FIG. 3 Measured spectral radiance and fitted graybody spectrum.



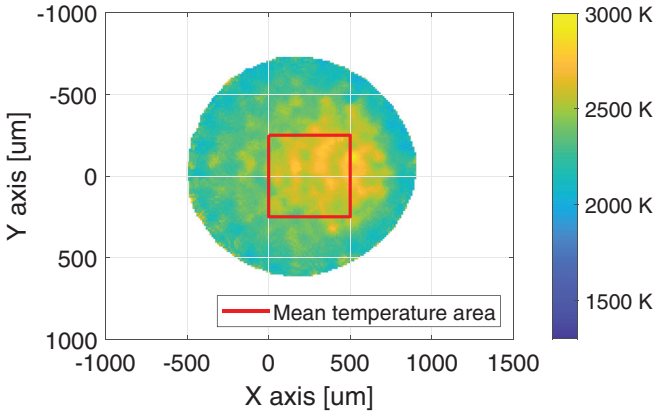
904 nm were also ignored out of empirical experience: they significantly diverged from the graybody spectrum and increased significantly the minimal fitting error.

Figure 3 shows an example of a pixel's measured spectral radiance and the corresponding fitted graybody spectrum. In this example, the graybody spectrum shows very good agreement with the measurements. When the method is applied to each pixel of a hyperspectral image, a 2D temperature distribution is obtained (see fig. 4). The temperature estimation method was implemented in MATLAB and took approximatively 32 s to solve a 300×300 hyperspectral image on six cores (Intel Core i7-9850H). The required time to solve the minimization problem prevents this method from being used in a real-time control system given the time constant of the DED proces.[19]

Temperature Distribution and Extraction of the Relevant Metric

An example of temperature distribution obtained with the estimation method is shown in figure 4. The plot shows only temperatures at locations where the temperature estimation method could fit a graybody spectrum to the measurements. Locations where the signal was too weak were ignored and are plotted in white on the plot. One can notice than the minimum temperature computed in this example is above the melting temperature of 316L stainless steel (1,670 K). As a result, the outline of the melt pool cannot be detected from the temperature distribution. Indeed, the maximal radiance of the blackbody spectrum is proportional to T^5. Therefore, the temperature range of the measurements is limited by the dynamic range of the camera. In other words, it is difficult to capture enough light from the areas at lower temperatures without saturating the pixels capturing light from the areas at higher temperatures.

FIG. 4 Temperature distribution ($P = 600$ W; $v = 800$ mm/min; $\dot{m} = 0$ g/min) obtained by multicolor pyrometry.



It should be noted that a temperature over the boiling temperature of stainless steel is estimated in the center of the melt pool. There is significant literature stating that a hot vapor plume exists above the melt pool.[20,21] This vapor plume could influence the camera measurements and lead to a higher temperature estimation in the center of the melt pool. Fortunately, the vapor plume could serve as a proxy to the melt pool because machine-learning (ML) algorithms do not need an understanding of the underlying process to build a model out of the data.

To use this temperature distribution with shallow data-driven algorithms, a global metric must be extracted. To avoid the effect of local noise that could result in a very high or low local temperature, we decided to use the mean temperature T_{mean} over an arbitrary area of the measured melt pool. The area is pictured as a dark rectangle in **figure 4**.

Prediction of the Melt Pool Temperature

SUPERVISED-LEARNING METHODS

The target relationship that is approximated by the algorithms is the following:

$$(P, v, \dot{m}) \rightarrow T_{mean} \tag{3}$$

The inputs of the models are therefore the laser power P, the laser speed v, and the powder feed rate $\dot{m}$, and the output is the mean temperature T_{mean} as defined in the previous subsection. This link between process parameters and the mean temperature has many applications in controlling the DED process.

To build a mathematical model of this target relationship, SL regression algorithms are used because of the continuous nature of the output variable. As explained in the introduction, a trial-and-error step is necessary to determine which algorithm is the most adapted to the phenomenon to be modeled. In this paper, we consider a selection of the most usual ML algorithms.[22] The algorithms and their hyperparameters are listed in **table 2**. The capacity and computational cost of each algorithm are briefly discussed in this subsection.

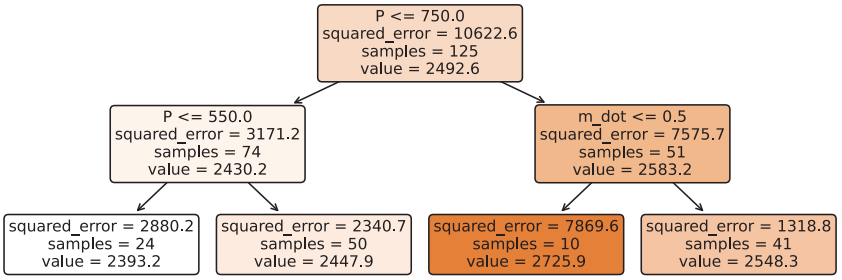
The usual linear regression model is used as a baseline to compare its performance with algorithms with more capacity. It is expected to underfit complex (nonlinear) relationships given its limited capacity. Two common variations of the linear regression are also included in the study: the ridge and the lasso regression. Ridge regression uses L2-norm regularization on the linear weights to prevent overfitting on the training set, and lasso regression uses L1-norm regularization to achieve both variable selection and overfitting prevention.[22]

Two nonparametric models are considered in this study: the K-nearest neighbor (KNN) and decision-tree (DT) algorithms. The KNN algorithm outputs a weighted averaged of the k closest training data points to the input, and the DT algorithm builds a flowchart-like structure based on the value of the features to make a prediction (see **fig. 5**). As nonparametric models, they can achieve high capacity, but their computational cost grows with the size of the training set.[23]

TABLE 2 Supervised-learning algorithms

Model	Hyperparameters	Value
Linear regression	—	—
Lasso	L1-regularization strength	[1, 10, 100]
Ridge	L2-regularization strength	[1, 10, 100]
K-nearest neighbors	Number of neighbors	[3, 5, 10]
	Weights	[Uniform, distance]
Decision-tree regression	Criterion	[MSE, Poisson]
	Maximum depth	[3, 6, 9, 12]
	Minimum number of samples at leaf node	[3, 5]
Support vector regression	Kernel function	[RBF, polynomial]
	Polynomial degree	[2, 3, 4]
	Kernel coefficient	[1, 10]
	Gamma	[1, 2]
Artificial neural network	Number of hidden layers	1
	Type of layer	Dense
	Number of nodes per layer	4

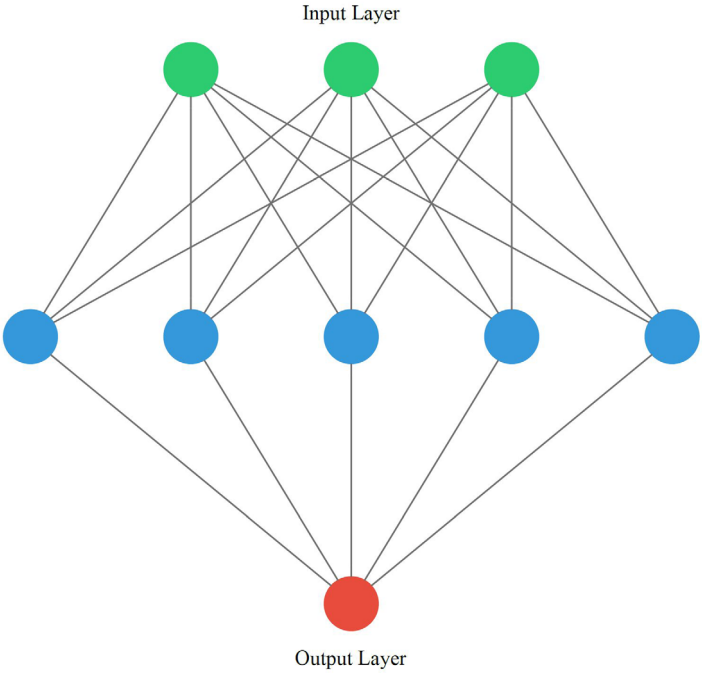
FIG. 5 Structure of a simple decision tree (maximum depth = 2).



A support vector regression (SVR) model is included in this study as an efficient and high-capacity SL algorithm. Because of its kernel trick it is capable of fitting a nonlinear model as a function of its features using convex optimization techniques that converge efficiently.[23]

Finally, a simple example of an artificial neural network (ANN) is also included in this study. A neural network is made of several neurons that are arranged in layers and connected from one layer to the next (see **fig. 6**). The output of each neuron is determined by a nonlinear function of the sum of its inputs. As a result, an ANN typically has a higher capacity than classic ML algorithms but requires a large data set to avoid underfitting and significant computational power to be trained.[23]

FIG. 6 Structure of a simple artificial neural network.



CROSS-VALIDATION AND HYPERPARAMETER OPTIMIZATION

Hyperparameter optimization was implemented to maximize the performance of an ML algorithm on the particular data set introduced earlier. Each considered hyperparameter value is listed in table 2. To properly evaluate the generalization error and prevent data leakage during the hyperparameter optimization, a nested k-fold cross-validation scheme was performed. First, we split the data set in k_1 times and trained and tested the model on each of the (training set, testing set) folds. The training set was used to train the model and tune the hyperparameters, and the testing set was used only to estimate the performance of the model. This step is called outer cross-validation.

During the evaluation of each of the k_1 pairs, the training set was then further split k_2 times in (training set, validation set) folds. The training set was used to actually train the model, and the validation set was used to evaluate the performance of a particular choice of hyperparameters. This step is called inner cross-validation.

To summarize, all of the data points are once part of the test set in one of the outer folds. The overall performance of a given model is given by the mean and standard deviation of the k_1 estimations of the generalization error over the k_1 outer folds.

The two metrics that we chose to compute the generalization error of this SL problem are the R^2 score and the root mean square error (RMSE). The R^2 score can be interpreted as the percentage of variance of the testing set that is captured by the model. A value of 1 means that the model perfectly represents the testing set.

$$R^2 = 1 - \frac{\sum_{n=1}^{N} \left(\hat{y}_n - y_n\right)^2}{\sum_{n=1}^{N} \left(\bar{y} - y_n\right)^2} \tag{4}$$

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N} \left(\hat{y}_n - y_n\right)^2}{N}} \tag{5}$$

where:

N = number of values in the set,
y_n = n^{th} value in the set,
$\hat{y}_n$ = algorithm predicted value, and
$\bar{y}$ = mean value of the set.

RESULTS AND DISCUSSION

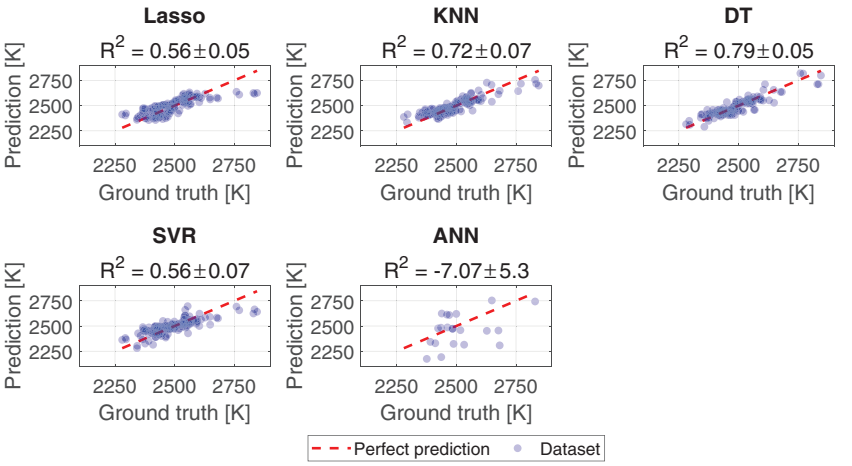
The algorithms presented in the previous subsection were implemented with the scikit-learn[24] and Keras Python packages. They were fitted to the simulated data set on a Core i7-9850H laptop with nested cross-validation for the ML algorithms (k_1 = 5 and k_2 = 4) and simple cross-validation (k_1 = 5) with 500 epochs per fit for the ANN. The nested cross-validation scheme was not applied to the ANN to limit the computational cost. The mean and standard deviation of the metrics were computed out of the five values of the five-fold outer cross-validation for the ML algorithms.

The resulting metrics and training times are listed in table 3, and corresponding regression plots are shown in figure 7. We can observe that the DT regression is the most effective model. The KNN, linear regression, and SVR have a good to average

TABLE 3 Metrics of supervised-learning algorithms fitted to the experimental data set

Model	R^2		RMSE (K)		Training Time (ms)
	Mean	SD	Mean	SD	
Linear regression	0.53	0.08	65.1	13.2	3.93
Lasso	0.56	0.05	64.0	14.8	3.85
Ridge	0.55	0.05	64.7	15.4	3.85
KNN	0.72	0.07	49.5	7.4	4.08
DT regression	0.79	0.05	44.0	10.9	4.45
SVR	0.56	0.07	64.3	19.6	7.2
ANN	−7.07	5.30	248.2	29.5	141.6

FIG. 7 Regression plots for several SL algorithms.



performance, and the ANN has a very poor performance. These results show that the choice of algorithm heavily influences the performance of the model.

We can also notice that the hyperparameter optimization of the linear regression did not lead to a significant performance improvement for the ridge and lasso regressions. This means that the regularization techniques in those algorithms are not well adapted to the data set in this study.

The ANN is generally considered more able than traditional ML algorithms to generalize high-dimensional data.[23] Despite this, the ANN fitted to the simulated data set has a much lower R^2 score and much higher RSME than all of the other algorithms. One possible explanation could be the relatively limited size of the data set. Indeed, the downside of a much more plastic model such as the ANN is that it requires much more data to fit all of its internal weights efficiently and avoid a high model variance. Another possible explanation is that the structure of the considered ANN is not appropriate for modeling the target relationship. To investigate these possibilities, a larger data set should be produced, and several ANN structures should be evaluated.

Finally, the considered data set represents only the steady-state process because the collected values are averaged over the lengths of the tracks. This has limited practical applications and requires a specific experimental campaign because reaching steady-state is not common while printing a normal part with varying geometrical features. In the future, transient data and therefore the influence of the temperature history should be considered for applications in the control and condition monitoring of the process. The increase in input variables (features) should require deeper algorithms, which in turn require a larger data set to be successfully trained, but it should be possible to collect training data during normal prints.

Conclusion

In this paper, an experimental in situ monitoring data set of the DED process was presented, and the postprocess method was described. The postprocess method consists of a temperature estimation method that makes use of steady-state hyperspectral optical data. The resulting experimental data set is then used in conjunction with SL methods to correlate the steady-state mean temperature over an arbitrary surface of the melt pool and the process parameters.

The temperature estimation method used in this work allows a refined temperature distribution of the melt pool to be computed. However, it suffered from several limitations. The range of temperature that the method computes does not cover the full liquid area of the melt pool because of the limited dynamic range of the camera. In addition, this method requires heavy computational power and a significant computing time, which limits its possibilities in real-time applications.

The performance of SL methods was compared, and it was shown that the DT regression could efficiently approximate the relationship between mean temperature and process parameters. This result suggests that the choice of algorithm heavily influences the performance of the overall model. The performance of the ANN was particularly low and might either require more data or a different structure.

For future work, the temperature estimation could be improved by capturing light from the melt pool at several exposures to increase the dynamic range of the camera. Transient data and the influence of the temperature history should be considered for applications in the control and condition monitoring of the process. The increase in input variables (features) should require deeper algorithms, which in turn require a larger data set to be successfully trained.

ACKNOWLEDGMENTS

The authors would like to acknowledge the support of the FWO HIPAS project (Grant S009319N1). In addition, Jan Helsen received funding from the Flemish government (AI Research Program).

References

1. *Standard Terminology for Additive Manufacturing—General Principles—Terminology*, ISO/ASTM 52900:2015 (Geneva, Switzerland: International Organization for Standardization, 2015).
2. W. Yan, S. Lin, O. L. Kafka, Y. Lian, C. Yu, Z. Liu, J. Yan et al., "Data-Driven Multi-Scale Multi-Physics Models to Derive Process–Structure–Property Relationships for Additive Manufacturing," *Computational Mechanics* 61 (2018): 521–541.
3. C. Knaak, L. Masseling, E. Duong, P. Abels, and A. Gillner, "Improving Build Quality in Laser Powder Bed Fusion Using High Dynamic Range Imaging and Model-Based Reinforcement Learning," *IEEE Access* 9 (2021): 55214–55231.
4. W.-W. Liu, Z.-J. Tang, X.-Y. Liu, H.-J. Wang, and H.-C. Zhang, "A Review on In-Situ Monitoring and Adaptive Control Technology for Laser Cladding Remanufacturing," *Procedia CIRP* 61 (2017): 235–240.

5. Z.-J. Tang, W.-W. Liu, Y.-W. Wang, K. M. Saleheen, Z.-C. Liu, S.-T. Peng, Z. Zhang, and H.-C. Zhang, "A Review on In Situ Monitoring Technology for Directed Energy Deposition of Metals," *International Journal of Advanced Manufacturing Technology* 108 (2020): 3437–3463.
6. W. Devesse, D. De Baere, and P. Guillaume, "High Resolution Temperature Measurement of Liquid Stainless Steel Using Hyperspectral Imaging," *Sensors* 17, no. 1 (2017): 91, https://doi.org/10.3390/s17010091
7. Y. Huang, M. B. Khamesee, and E. Toyserkani, "A New Physics-Based Model for Laser Directed Energy Deposition (Powder-Fed Additive Manufacturing): From Single-Track to Multi-Track and Multi-Layer," *Optics & Laser Technology* 109 (2019): 584–599.
8. X. Li, S. Siahpour, J. Lee, Y. Wang, and J. Shi, "Deep Learning-Based Intelligent Process Monitoring of Directed Energy Deposition in Additive Manufacturing with Thermal Images," *Procedia Manufacturing* 48 (2020): 643–649.
9. M. Khanzadeh, S. Chowdhury, M. Marufuzzaman, M. A. Tschopp, and L. Bian, "Porosity Prediction: Supervised-Learning of Thermal History for Direct Laser Deposition," *Journal of Manufacturing Systems* 47 (2018): 69–82.
10. M. Khanzadeh, S. Chowdhury, M. A. Tschopp, H. R. Doude, M. Marufuzzaman, and L. Bian, "In-Situ Monitoring of Melt Pool Images for Porosity Prediction in Directed Energy Deposition Processes," *IISE Transactions* 51 (2019): 437–455.
11. F. Caiazzo and A. Caggiano, "Laser Direct Metal Deposition of 2024 Al Alloy: Trace Geometry Prediction via Machine Learning," *Materials* 11, no. 3 (2018): 444, https://doi.org/10.3390/ma11030444
12. Z. Zhang, Z. Liu, and D. Wu, "Prediction of Melt Pool Temperature in Directed Energy Deposition Using Machine Learning," *Additive Manufacturing* 37 (2021): 101692, https://doi.org/10.1016/j.addma.2020.101692
13. D. R. Feenstra, A. Molotnikov, and N. Birbilis, "Utilisation of Artificial Neural Networks to Rationalise Processing Windows in Directed Energy Deposition Applications," *Materials & Design* 198 (2021): 109342, https://doi.org/10.1016/j.matdes.2020.109342
14. C. Snyers, J. Ertveldt, J. Sanchez-Medina, Z. Jardon, and J. Helsen, "Prediction of Build Geometry for DED Using Supervised Learning Methods on Simulated Process Monitoring Data," *Journal of Laser Applications* 33, no. 4 (2021): 042052, https://doi.org/10.2351/7.0000526
15. J. Ertveldt, P. Guillaume, and J. Helsen, "MiCLAD as a Platform for Real-Time Monitoring and Machine Learning in Laser Metal Deposition," *Procedia CIRP* 94 (2020): 456–461.
16. M. Izadi, A. Farzaneh, M. Mohammed, I. Gibson, and B. Rolfe, "A Review of Laser Engineered Net Shaping (LENS) Build and Process Parameters of Metallic Parts," *Rapid Prototyping Journal* 26, no. 6 (2020): 1059–1078.
17. B. Zheng, J. C. Haley, N. Yang, J. Yee, K. W. Terrassa, Y. Zhou, E. J. Lavernia, and J. M. Schoenung, "On the Evolution of Microstructure and Defect Control in 316L SS Components Fabricated via Directed Energy Deposition," *Materials Science and Engineering: A* 764 (2019): 138243, https://doi.org/10.1016/j.msea.2019.138243
18. W. Devesse, "Development and Validation of Innovative Modeling, Monitoring and Control Strategies for 3D Metal Printing" (PhD thesis, Vrije Universiteit Brussel, 2017).
19. M. Akbari and R. Kovacevic, "Closed Loop Control of Melt Pool Width in Robotized Laser Powder–Directed Energy Deposition Process," *The International Journal of Advanced Manufacturing Technology* 104 (2019): 2887–2898.
20. D. D. Baere, W. Devesse, J. Helsen, and P. Guillaume, "Spatial Distributed Spectroscopic Monitoring of Melt Pool and Vapor Plume during the Laser Metal Deposition Process," *Procedia CIRP* 94 (2020): 445–450.

21. A. R. Nassar, B. Starr, and E. W. Reutzel, "Process Monitoring of Directed-Energy Deposition of Inconel-718 via Plume Imaging," in *2015 International Solid Freeform Fabrication Symposium* (Austin, TX: University of Texas at Austin, 2015), 284–294.
22. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning* (New York: Springer, 2009).
23. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (Cambridge, MA: MIT Press, 2016).
24. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel et al., "Scikit-Learn: Machine Learning in Python," *Journal of Machine Learning Research* 12 (2011): 2825–2830.